04-20-00

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P. O. Box 272400
Fort Collins, Colorado 80528-9599

PATENT APPLICATION

ATTORNEY DOCKET NO. 10991401-1

A

04/19/00 U.S. PTO

IN THE U.S. PATENT AND TRADEMARK OFFICE
Patent Application Transmittal Letter

jc525 U.S. PTO 09/552060 04/19/00

ASSISTANT COMMISSIONER FOR PATENTS
Washington, D.C. 20231

Sir:

Transmitted herewith for filing under 37 CFR 1.53(b) is a(n): (X) Utility ( ) Design
(X) original patent application,
( ) continuation-in-part application

INVENTOR(S): **Peter J Lange et al**

TITLE: **A Constant Size Image Display Independent Of Screen Resolution**

Enclosed are:

(X) The Declaration and Power of Attorney. (X) signed ( ) unsigned or partially signed

(X) 2 sheets of drawings (one set) ( ) Associate Power of Attorney

( ) Form PTO-1449 ( ) Information Disclosure Statement and Form PTO-1449

( ) Priority document(s) ( )(Other) _____ (fee $_____)

CLAIMS AS FILED BY OTHER THAN A SMALL ENTITY				
(1) FOR	(2) NUMBER FILED	(3) NUMBER EXTRA	(4) RATE	(5) TOTALS
TOTAL CLAIMS	5 — 20	0	X $18	$ 0
INDEPENDENT CLAIMS	5 — 3	2	X $78	$ 156
ANY MULTIPLE DEPENDENT CLAIMS	0		$260	$ 0
BASIC FEE: Design $310.00); Utility $690.00)				$ 690
			TOTAL FILING FEE	$ 846
			OTHER FEES	$
			TOTAL CHARGES TO DEPOSIT ACCOUNT	$ 846

Charge $ 846 to Deposit Account 08-2025. At any time during the pendency of this application, please charge any fees required or credit any over payment to Deposit Account 08-2025 pursuant to 37 CFR 1.25. Additionally please charge any fees to Deposit Account 08-2025 under 37 CFR 1.16, 1.17,1.19, 1.20 and 1.21. A duplicate copy of this sheet is enclosed.

"Express Mail" label no. EL483353106US

Date of Deposit 4/19/00

I hereby certify that this is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to: Assistant Commissioner for Patents, Washington, D.C. 20231.

By Laura M Clark

Typed Name: **Laura M. Clark**

Respectfully submitted,

Peter J Lange et al

By _____

Steven L Webb

Attorney/Agent for Applicant(s)
Reg. No. **44,395**
Date: 4/19/00

Telephone No.: **(970) 898-7745**

Rev 12/99 (TransNew)

- Attach as First Page to Transmitted Papers -

A constant size image display independent of screen resolution

FIELD OF THE INVENTION

The present invention relates generally to web pages and more specifically to a web page display that maintains the size of an image being displayed independent of the physical screen size and the resolution of the display.

BACKGROUND OF THE INVENTION

Today computer display devices have a wide range of available screen resolutions. Some of the standard resolutions are 640x480, 800x600, 1024x768, and 1152x768. Each of these resolutions gives the number of pixels that are displayed in the width and height of the display area. For example the 640x480 resolution displays 640 pixels across the width of the display and 480 pixels across the height of the display. These screen resolutions are typically independent of the actual size of the display area. For example a 15-inch monitor can use a resolution of 1024x768 and a 21-inch monitor can use a resolution of 640x480.

The display driver is the software running on the computer that knows about the hardware of the display device. The display driver typically knows the physical size of the display, the refresh rate of the display, the amount of video memory in the display device, the bit depth or number of colors that can be displayed, and the supported resolutions for the display device. These settings, including the resolution used for the display, can typically be changed by the user by adjusting the display driver.

Web browsers, for example Microsoft's Internet Explorer ®, display web pages. The web page, when loaded, tells the web browser how and what to display.

Typically this description of what the web page should look like is stored in a web page description language like HyperText Markup Language (HTML). The HTML document that describes a web page typically defines where all elements of the web page should be located. Web pages can contain text elements, graphic elements, as well as video and audio elements. The graphic elements can be bitmap images and vector objects like drawing. There are a number of parameters that can be specified when using HTML to display an image. Some of the parameters are 1) the image starting location 2) the way the text wraps the image 3) the height and width of the image. Currently there are two ways to display an image using HTML.

The first method is to specify a percent of the width and height of the web browser window. The second method is to specify the number of pixels for the width and height of the image. Each of these methods can cause problems when displaying images on different sized displays or same sized displays using different resolutions.

Displaying an image with HTML by specifying the percentage of the web browser window can cause the image to be non-proportionally scaled. Only integer values of the percent of the height and width of the web browser are available for use. When the web browser's height and width do not scale to the image height and width by an integer value, the image will be scaled by a different value in the height and width. The stretched or distorted non-proportionally scaled image can diminish the impact of using an image in the web page. Precise control over the layout of the page to be displayed cannot be achieved because the integer value requirement limits the height and width control to one percent of the height and width of the web browser.

Even when the image is an integral size of the web browser for a given web browser size, the web browser does not maintain a constant size or a constant width to height ratio. For example when the web browser is changed from "normal" view to

"full screen" view the web browser typically changes its display aspect ratio. The aspect ratio is the ratio of the width to the height of a web page.

Displaying an image in a web page by specifying the number of pixels for the image height and width allows the image to remain the same size when the web browser is resized. But the size of the image is different depending on the current resolution of the display driver. For example an image that is 4x4 inches on a 17-inch monitor using a resolution of 1280x1024 would be 8x8 inches on the same monitor when using a resolution of 640x480. A potential difference of 2 to 1 in image sizes makes it difficult to create a page layout by specifying the image size using the number of pixels for the height and width. Having an image displayed at half the size that the user desired makes the image much harder to see and causes the image to lose impact.

Accordingly there is a need for a web page display that maintains the size of an image being displayed independent of the physical screen size and the resolution of the display.

SUMMARY OF THE INVENTION

The present invention is a method and apparatus for displaying images on web pages at a constant size or at a constant proportion of the screen size, independent of screen resolution and independent of physical screen size. By determining the current screen resolution and size and then scaling the image, the correct image size can be maintained.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a flow chart of a method to display an image, embedded in a web page, at a desired size in accordance with the present invention.

Figure 2 is a flow chart of a method to display an image, embedded in a web page, at a fixed percent of the width of the display window using a constant aspect ratio in accordance with the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

The present invention is a method and apparatus for displaying images on web pages at a constant size or at a constant proportion of the screen size, independent of screen resolution and independent of physical screen size. By determining the current screen resolution and size and then scaling the image, the correct image size can be maintained.

The main problem today is that a page that is described using HTML, to be displayed on a web browser, does not have visibility of the display driver. Without visibility of the display driver the HTML page is unable to access the current settings of the display drive. These settings include the current display resolution.

A Java applet can be embedded into the HTML description of the page to be displayed. A Java applet is like a program. When a web browser is reading an HTML description of a page and encounters a Java applet, program control is passed to the Java applet. The Java applet executes and then passes program control back to the web browser. Java applets have visibility of the display driver. This allows Java applet to determine the current settings of the display driver.

Figure 1 is a flow chart showing the steps the Java applet uses to display an image at a desired size. The Java applet retrieves the physical display size (102), then

reads the image resolution (104). The applet then gets the current settings of the display driver (106). Using the information from the previous steps the applet calculates the physical size of the image (108) and the physical resolution of the screen (110). Using the display attributes and the desired image size, the correct scaling for the image to be displayed can be calculated (112). The Java applet can then display the image at the correct size (114).

The code for the constant size image embodiment of the current invention is included in appendix A. Page 12, line 8 is the HTML source for the page to be displayed. On line 12 of page 12 the Java applet is embedded into the HTML page. The HTML page also contains the name and path to the image to be displayed (line 20, page 12). For this example embodiment the physical height and width of the screen is embedded into the HTML source (page 12, line 21). In the actual operation of the invention the physical height and width of the screen would be input by the user and stored in a file at a known location. If the Java applet did not detect the file the applet would ask the user for this information and then store the physical size and width at the known location.

The code for the Java applet that is launched by the HTML page starts at line 4 on page 9. The Java applet reads the image data and physical height and width from the HTML page (lines 26 – 40 of page 9). The applet then creates a container where it will display the image (line 45 page 9). The applet then launches a second Java applet called CSPanel (line 1 page 10). The code for the CSPanel applet starts on line 10 of page 10. The CSPanel applet reads the x and y resolution of the image from the header of the JPEG image file. Many image files used on the Internet are currently stored in the JPEG file format, however in the actual embodiment of the current

invention a number of file types could be supported. A JPEG file was used in the example embodiment for clarity of understanding.

The "paintComponent" function of the example embodiment starts on page 11, line 16 of appendix A. The "paintComponent" function in a program operating in the Microsoft Windows ® environment is responsible for controlling the display area that the program is using. The operating system sends a paintComponent request to the program each time that the display area needs to be refreshed. For example, a paintComponent request is sent to the program when the program window is resized or moved.

Each time the paintComponent function is called the logical screen resolution is retrieved from the display driver (line 22 – 23 page 11). The physical size of the image is calculated on lines 26 – 29 of page 11. The size of the image for the example embodiment is based on the scanned resolution of the image. In the actual embodiment a new field can be added to the image header file to store a desired image display size. For example, display this image 4.5 inches in width. This new field could be used instead of the scanned image resolution, allowing more control over the displayed size of the image.

The physical resolution of the screen is calculated on lines 32 – 33 of page 11. This resolution is what changes when the user switches between 640x480 and 1024x768 on the same physical screen. Using the physical resolution of the screen the logical width and height of the image is calculated (lines 36,37 of page 11). The logical width and height is used to draw the image to the screen at the correct size independent of screen resolution or physical screen size (line 49 of page 11).

For web page layout design it may be desirable to display an image at a constant percent of the web browser window width or height, while maintaining a

constant image aspect ratio. A second embodiment of the current invention would display an image at a constant percent of the web browser window width. In this embodiment the physical screen height and width are not used. The desired image display width (206) is the total display width of the web browser times the percentage of the display width to be used by the image. The scale factor would be used to scale the image (208) in both the width and height to maintain a constant image aspect ratio. For example, when the desired image display width is 80 percent of the browser window width, the browser window width is 1000 pixels, and the image is 400 pixels wide, the scale factor would be 400/(1000 * .80) = ½. When the image display width is set as a percent of the browser window display width, the image will not be re-scaled when the web browser window height is changed. The image size can be controlled using either a percent of the web browser window width or a percent of the web browser window height.

Appendix B contains the code for an example embodiment of a Java applet that displays an image in a web page at a constant percent of the browser window in accordance with the present invention. The source for the HTML page starts at line 18 of page 15. Embedded into this page is the name of the image (line 33 page 15), the path to the image (line 33 page 15), and the percent of the page width (line 33 page 15) to display the image. When this page is displayed it launches the embedded Java applet TestApplet (line 30 page 15). The code for TestApplet starts on line 4 of page 13. TestApplet reads the image and percent of width information from the HTML page and then launches a second Java applet called TestPanel (line 39 page 13). The code for the Java applet TestPanel starts on line 46 of page 14. TestApplet contains the paintCompnent that is called by the operation system each time the screen needs to be refreshed. TestPanel scales the image to the percent of the page width or the

percent of the page height which ever one is set (lines 35 – 50 page 14) and then displays the image at the correct size (lines 5 – 12 page 15).

The foregoing description of the present invention has been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise form disclosed, and other modifications and variations may be possible in light of the above teachings. For example, any programmatic control that has visibility of the display driver could be used instead of the Java applet. The embodiment was chosen and described in order to best explain the principles of the invention and its practical application to thereby enable others skilled in the art to best utilize the invention in various embodiments and various modifications as are suited to the particular use contemplated. It is intended that the appended claims be construed to include other alternative embodiments of the invention except insofar as limited by the prior art.

CLAIMS

What is claimed is:

1. A method of displaying an image embedded in a web page, comprising the steps of:

determining the physical size of the display and the current resolution of the display;

determining the desired physical size of the image to be displayed;

displaying the image at the desired physical size.

2. A system for displaying an image embedded in a web page, comprising:

an image stored in the system, the image having a desired physical size;

a display having a physical display area and a current resolution;

a processing system, the processing system comprising at least one processor, wherein the processing system scales the image, using the physical display area and the current resolution, such that the image is displayed at the desired physical size.

3. A system for displaying an image embedded in a web page, comprising:

means for determining the physical size and current resolution of the display;

means for reading an image to be displayed;

means for displaying the image at a desired size.

4. A method of displaying an image embedded in a web page, comprising the steps of:

determining the desired length, in one dimension, of a browser window to be used to display the image;

scaling the image, using a constant aspect ratio, to match the desired length;

displaying the image.

5. A system for displaying an image embedded in a web page, comprising:

an image, the image having a desired size relative to the size of a browser used to display the image;

a display having a current resolution;

a processing system, the processing system comprising at least one processor, wherein the processing system scales the image, using the current resolution, such that the image is displayed, using a constant aspect ratio, at the desired relative size.

ABSTRACT OF THE DISCLOSURE

A method and apparatus for displaying images on web pages at a constant size, independent of screen resolution and independent of physical screen size is disclosed. By determining the current screen resolution and size and then scaling the image, the image can be displayed at the correct size.

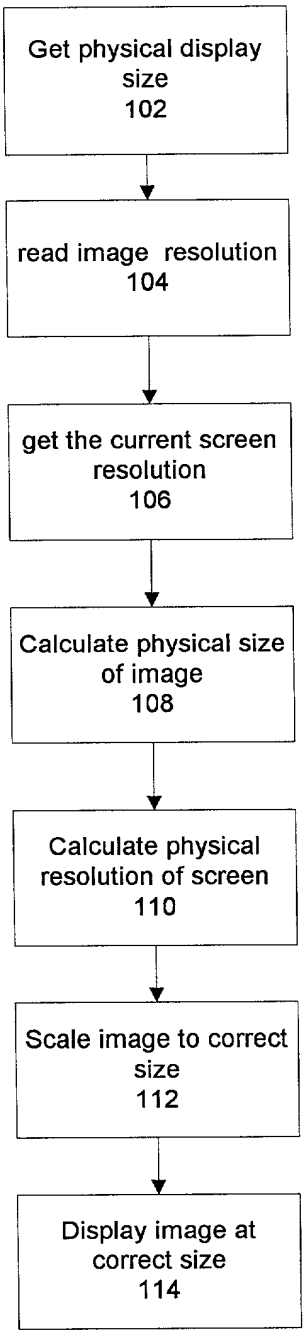
Get physical display size
102
read image resolution
104
get the current screen resolution
106
Calculate physical size of image
108
Calculate physical resolution of screen
110
Scale image to correct size
112
Display image at correct size
114

Figure 1

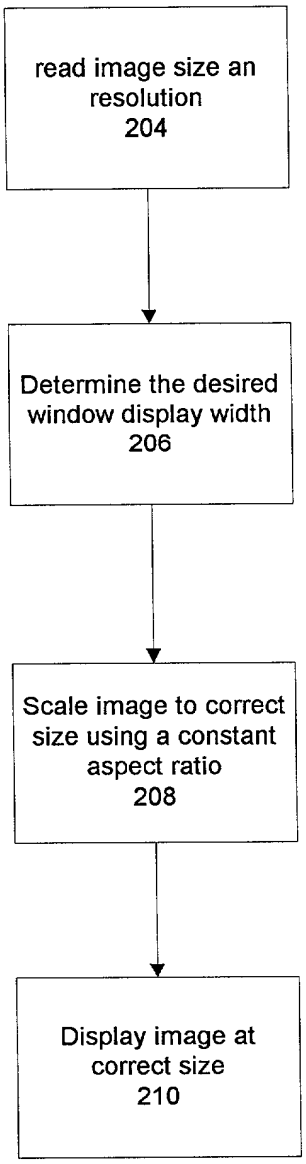
read image size an resolution
204
Determine the desired window display width
206
Scale image to correct size using a constant aspect ratio
208
Display image at correct size
210

Figure 2

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION

ATTORNEY DOCKET NO. 10991401-1

As a below named inventor, I hereby declare that:

My residence/post office address and citizenship are as stated below next to my name;

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled:

A Constant Size Image Display Independent Of Screen Resolution

the specification of which is attached hereto unless the following box is checked:

( ) was filed on _____ as US Application Serial No. or PCT International Application Number _____ and was amended on _____ (if applicable).

I hereby state that I have reviewed and understood the contents of the above-identified specification, including the claims, as amended by any amendment(s) referred to above. I acknowledge the duty to disclose all information which is material to patentability as defined in 37 CFR 1.56.

Foreign Application(s) and/or Claim of Foreign Priority

I hereby claim foreign priority benefits under Title 35, United States Code Section 119 of any foreign application(s) for patent or inventor(s) certificate listed below and have also identified below any foreign application for patent or inventor(s) certificate having a filing date before that of the application on which priority is claimed:

COUNTRY	APPLICATION NUMBER	DATE FILED	PRIORITY CLAIMED UNDER 35 U.S.C. 119
N/A			YES: ___ NO: ___
			YES ___ NO ___

Provisional Application

I hereby claim the benefit under Title 35, United States Code Section 119(e) of any United States provisional application(s) listed below:

APPLICATION SERIAL NUMBER	FILING DATE
N/A	

U. S. Priority Claim

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code Section 112, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, Section 1.56(a) which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

APPLICATION SERIAL NUMBER	FILING DATE	STATUS (patented/pending/abandoned)
N/A		

POWER OF ATTORNEY:

As a named inventor, I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and transact all business in the Patent and Trademark Office connected therewith:

Customer Number 022879

Place Customer Number Bar Code Label here

Send Correspondence to:
HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80528-9599

Direct Telephone Calls To:
Steven L Webb
(970) 898-7745

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full Name of Inventor: Peter J Lange **Citizenship:** US

Residence: 818 Locust Street Windsor CO 80550

Post Office Address: Same as residence

Inventor's Signature _____ Date 4-18-00

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION (continued)

ATTORNEY DOCKET NO. 10991401-1

Full Name of # 2 joint inventor: Robert E Chalstrom Citizenship: US

Residence: 1417 Hepplewhite Court Fort Collins CO 80526

Post Office Address: Same as residence

Robert E. Chalstrom — Inventor's Signature

4-18-00 — Date

Full Name of # 3 joint inventor: Melvin Brent Wilkins Citizenship: US

Residence: 2906 Wagonwheel Court Fort Collins, CO 80526

Post Office Address: Same as residence

Melvin Brent Wilkins — Inventor's Signature

4/18/2000 — Date

Full Name of # 4 joint inventor: _____ Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____ Date _____

Full Name of # 5 joint inventor: _____ Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____ Date _____

Full Name of # 6 joint inventor: _____ Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____ Date _____

Full Name of # 7 joint inventor: _____ Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____ Date _____

Full Name of # 8 joint inventor: _____ Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____ Date _____

Appendix A

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.io.*;

public class CSApplet extends JApplet
{
     private String image;
     private double physicalWidth;
     private double physicalHeight;
     public void init()
     {
          super.init();

          physicalWidth = 0.0;
          physicalHeight = 0.0;

          // Read in the values of the parameters from
the html page
          // The parameters are image, physicalWidth, and
physicalHeight

          image = getParameter("image");

          String param = getParameter("physicalWidth");

          if (param != null)
          {
               physicalWidth = Double.parseDouble(param);
          }

          param = getParameter("physicalHeight");

          if (param != null)
          {
               physicalHeight =
Double.parseDouble(param);
          }

          // Add a panel to the applet where the image
will be drawn
          Container contentPane = getContentPane();

          try
          {
               // Create the panel and pass it the html
parameter values
```

```
                contentPane.add(new CSPanel(image,
physicalWidth, physicalHeight));
            }
            catch (IOException e)
            {
            }
        }
}

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.*;
import java.io.*;

class CSPanel extends JPanel
{
        private String imageName;
        private double physicalWidth;
        private double physicalHeight;
        private int imageResX;
        private int imageResY;
        private Image image;
        CSPanel(String imageName, double physicalWidth,
double physicalHeight) throws IOException
        {
                this.imageName = imageName;
                this.physicalWidth = physicalWidth;
                this.physicalHeight = physicalHeight;

                // Hack for reading the xResolution and
yResolution from the JPEG header
                FileInputStream in = new FileInputStream(new
File(imageName));
                DataInputStream din = new DataInputStream(in);

                // Skip over the first seven shorts
                for (int i = 0; i < 7; i++)
                {
                        din.readShort();
                }

                // get the x/y resolution
                imageResX = din.readShort();
                imageResY = din.readShort();

                // close streams
                din.close();
                in.close();

                // read in the image
```

```
            image =
    Toolkit.getDefaultToolkit().getImage(imageName);

                // wait until the image is entirely read in
                MediaTracker tracker = new MediaTracker(this);
                tracker.addImage(image, 0);

                try
                {
                     tracker.waitForID(0);
                }
                catch(InterruptedException e)
                {
                }
          }
          public void paintComponent(Graphics g)
          {
                super.paintComponent(g);

                // get the logical screen resolution ie
    640x480, 1024x768
                Toolkit tk = Toolkit.getDefaultToolkit();
                Dimension screen = tk.getScreenSize();

                // compute the physical size of the image
                double xInches = image.getWidth(this) /
    (double) imageResX;
                double yInches = image.getHeight(this) /
    (double) imageResY;

                // compute the physical dpi of the display
                double xDPI = screen.width / physicalWidth;
                double yDPI = screen.height / physicalHeight;

                // compute the logical width of the image
                double newWidth = xDPI * xInches;
                double newHeight = yDPI *yInches;

                // setup the graphics context
                Graphics2D g2 = (Graphics2D) g;

          g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
    RenderingHints.VALUE_ANTIALIAS_ON);

          g2.setRenderingHint(RenderingHints.KEY_RENDERING,
    RenderingHints.VALUE_RENDER_QUALITY);

                // draw the image in the panel with scaling
                g2.drawImage(image, 0, 0, (int) newWidth, (int)
    newHeight, null);
          }
```

```
}

<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
<head>
  <meta name="GENERATOR" content="Mozilla/4.61 [en] (WinNT; I)
[Netscape]">
  <title>TestApplet1</title>
</head>
<body>
<object classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93"
   width="100%" height="100%" align="middle"
   codebase="http:/<object classid="clsid:8AD9C840-044E-11D1-B3E9-
00805F499D93"
   width="100%<param NAME="code" VALUE="CSApplet.class"><param
NAME="codebase" VALUE="classes/"><param NAME="type"
VALUE="application/x-java-applet;version=1.2.2"><param NAME="image"
value="d:/patent/moose.jpg"><param NAME="physicalWidth"
value="11.25"><param NAME="physicalHeight" value="8.5"><param
NAME="scriptable" VALUE="true">No
JDK 1.2 support for APPLET!!</object>
</body>
</html>
```

Appendix B

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class TestApplet extends JApplet
{
     private String image;
     private double percentOfWidth;
     private double percentOfHeight;
     public void init()
     {
          super.init();

          percentOfWidth = 0.0;
          percentOfHeight = 0.0;

          image = getParameter("image");

          String param = getParameter("percentOfWidth");

          if (param != null)
          {
               percentOfWidth =
Double.parseDouble(param);
          }

          param = getParameter("percentOfHeight");

          if (param != null)
          {
               percentOfHeight =
Double.parseDouble(param);
          }

          Container contentPane = getContentPane();
          contentPane.add(new TestPanel(image,
percentOfWidth, percentOfHeight));
     }
}


import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class TestPanel extends JPanel
{
```

```
        private String imageName;
        private double percentOfWidth;
        private double percentOfHeight;

        private Image image;
        TestPanel(String imageName, double percentOfWidth,
    double percentOfHeight)
        {
             this.imageName = imageName;
             this.percentOfWidth = percentOfWidth;
             this.percentOfHeight = percentOfHeight;

             image =
    Toolkit.getDefaultToolkit().getImage(imageName);

             MediaTracker tracker = new MediaTracker(this);
             tracker.addImage(image, 0);

             try
             {
                  tracker.waitForID(0);
             }
             catch(InterruptedException e)
             {
             }
        }
        public void paintComponent(Graphics g)
        {
             super.paintComponent(g);

             Dimension container = getSize();
             double newWidth;
             double newHeight;

             if (percentOfWidth != 0.0)
             {
                  // constant percentage of width
                  newWidth = container.width *
    percentOfWidth / 100.0;
                  newHeight = newWidth *
                            ((double) image.getHeight(this)
    / (double) image.getWidth(this));
             }
             else
             {
                  // constant percentage of height
                  newHeight = container.height *
    percentOfHeight / 100.0;
                  newWidth = newHeight *
                            ((double) image.getWidth(this)
    / (double) image.getHeight(this));
```

```
        }

            Graphics2D g2 = (Graphics2D) g;

    g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON);

    g2.setRenderingHint(RenderingHints.KEY_RENDERING,
RenderingHints.VALUE_RENDER_QUALITY);

            g2.drawImage(image, 0, 0, (int) newWidth, (int)
newHeight, null);
    }
}


<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
<head>
  <meta name="GENERATOR" content="Mozilla/4.61 [en] (WinNT; I)
[Netscape]">
  <title>TestApplet2</title>
</head>
<body>
<object classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93"
   width="100%" height="100%" align="middle"
   codebase="http:/<object classid="clsid:8AD9C840-044E-11D1-B3E9-
00805F499D93"
   width="100%<param NAME="code" VALUE="TestApplet.class"><param
NAME="codebase" VALUE="classes/"><param NAME="type"
VALUE="application/x-java-applet;version=1.2.2"><param NAME="image"
value="d:/patent/beans.jpg"><param NAME="percentOfWidth"
value="50.0"><param NAME="scriptable" VALUE="true">No
JDK 1.2 support for APPLET!!</object>
</body>
</html>
```